REMARKS

I. Introduction

In response to the Office Action dated March 24, 2006, claims 1, 9, 10, 14, 23, 27 and 36 have been amended. Claims 1-39 remain in the application. Re-examination and re-consideration of the application, as amended, is requested.

II. Statutory Subject Matter Rejections

On pages (2)-(3) of the Office Action, claims 14 and 27 were rejected under 35 U.S.C. §101 as being directed to non-statutory subject matter because the claims recite the term "logic," the Office Action interprets the term "logic" as non-functional descriptive material, and the Office Action asserts that "logic" is not necessarily executable instructions.

Applicants' attorney has amended claims 14 and 27 to overcome this rejection.

However, should issues still remain in this regard, Applicants' attorney requests that the Examiner indicate how the rejection can be overcome, in accordance with the directives of the Examination Guidelines for Computer-Related Inventions. See Guidelines II M.P.E.P. § 2106. Specifically, should it be necessary, the Applicants' attorney requests that the Examiner identify features of the invention that would render the claimed subject matter statutory if recited in the claim. See Guidelines IV, M.P.E.P. § 2106.

III. Non-Art Rejections

On pages (3)-(4) of the Office Action, claims 1, 5, 6, 8, 10, 12-14, 18, 19, 21, 23, 25-27, 31, 32, 34, 36, 38, and 39 were rejected under 35 U.S.C. §112, second paragraph, as being indefinite.

Regarding claims 1, 14 and 27, the Office Action asserts that these claims are confusing.

Applicants' attorney has amended claims 1, 14 and 27 to overcome these rejections.

Regarding claims 5, 6, 8, 17, 18, 21, 31, 32 and 34, the Office Action asserts that these claims contradict each other, and that it is unclear what is included or excluded by the recited limitations of the claims.

Applicants' attorney traverses these rejections. Claims 5, 6 and 8 (as well as claims 17, 18 and 21, and claims 31, 32 and 34) are separate and distinct dependent claims, and have no relation to each other, except that they depend from the same independent claims. Each of these dependent claims recite a different one of the three types of query that trigger the optimization of Applicants' invention. It is irrelevant and immaterial if the Office Action believes that the claims contradict

each other (which Applicants' attorney disputes). Moreover, the claims clearly recite what is included or excluded by the limitations of the claims. Consequently, Applicants' attorney requests withdrawal of these rejections.

Regarding claims 10, 12-13, 23, 25-26, 36 and 38-39, the Office Action asserts that these claims are also confusing and vague.

Applicants' attorney has amended the claims to overcome these rejections, but nonetheless traverses these rejections. Claims 10 and 12-13 (as well as claims 23 and 25-26, and claims 36 and 38-39) recite the following:

> 10. The method of claim 1, wherein pruning logic determines whether the combined predicates are contradictory.
>
> 12. The method of claim 11, wherein no join is generated if the combined predicates are contradictory.
>
> 13. The method of claim 11, wherein the join is generated if the combined predicates are not contradictory.

The claims clearly recite their intended functions, namely that "pruning logic determines whether the combined predicates are contradictory" (claims 10, 23 and 36), and then "no join is generated if the combined predicates are contradictory" (claims 12, 25 and 38) or "a join is generated if the combined predicates are not contradictory (claims 13, 26 and 39). There is nothing confusing and vague about these limitations. Consequently, Applicants' attorney requests withdrawal of these rejections.

Regarding claim 27, the Office Action asserts that the term "an article of manufacture" renders the claim indefinite.

Applicants' attorney has amended claim 27 to overcome this rejection.

IV. <u>Prior Art Rejections</u>

A. <u>The Office Action Rejections</u>

On pages (4)-(7) of the Office Action, claims 1-5, 10-18, 23-31, and 36-39 were rejected under 35 U.S.C. §102(b) as being anticipated by Cheng, U.S. Patent No. 5,963,933. On pages (7)-(8) of the Office Action, claims 6-9, 19-22, and 32-35 were rejected under 35 U.S.C. §103(a) as being unpatentable over Cheng, in view of Jou, U.S. Patent No. 5,822,750.

Applicants' attorney respectfully traverses these rejections.

B. The Applicants' Independent Claims

Independent claims 1, 14 and 27 are directed to a method, system and article of manufacture for optimizing a query. Claim 1 is representative and recites a method of optimizing a query in a computer system, the query being performed by the computer system to retrieve data from a database stored on the computer system, the method comprising:

(a) combining join predicates from a query with local predicates from each branch of one or more UNION ALL views referenced by the query;

(b) analyzing the combined predicates; and

(c) generating the join, except when the analyzing step indicates that the combined predicates are always FALSE and the join generates an empty result.

C. The Cheng Reference

Cheng describes an efficient implementation of full outer join and anti-join. It is also a method for specifying SQL "full outer joins" which enables the use of arbitrary join conditions in specifying the query. This is enabled by equating a full outer join with a left outer join unioned with a right outer join less the matched tuples from the right outer join. A number of embodiments further present improvements in execution speed. One such improvement utilizes as a second operand of the union query a novel operator, the "ANTI-JOIN". The anti-join is implemented as a right outer join coupled with an ISNULL predicate.

D. The Jou Reference

Jou describes a relational database management system (RDBMS) that efficiently evaluates correlated subqueries by decorrelating them and taking care of the so-called SQL count bug and yet avoid using the expensive outer join operation. When there is no tuple match from a correlated subquery, the RDBMS query processor returns a tuple of null(s) from a scalar derived table, and then uses COALESCE function to generate a proper count value of zero from the null. The correlation level remains one. The query processor also performs a pass-through optimization to eliminate a join operation for floating SELECT operation by removing a join operation involving the magic operation, so that the correlation bindings are received from the correlation source table rather than the magic operation.

E. Applicants' Claims Are Patentable Over The References

Applicants' invention, as recited in independent claims 1, 14 and 27, is patentable over the Cheng and Jou references, because the claims recite limitations not found in the references.

Nonetheless, according to the Office Action, Cheng teaches all the elements of Applicants' independent claims at col. 8, line 37 to col. 10, line 25, col. 5, lines 45-56, col. 9, lines 20-35, col. 9, lines 32-34 and col. 11, lines 20-25.

These portions of Cheng are reproduced below:

> Cheng: Col. 8, line 35 – col. 10, line 25
>
> A first preferred embodiment of the present invention devises, for the first time, a mechanism which enables a query implementing a full outer join with any join conditions, including but not necessarily limited to the inequality predicate and disjunction. This is enabled by equating, in a novel manner, a full outer join with a left outer join unioned with a right outer join less the matched tuples from the right outer join. Referring to FIG. 4, a flow diagram of this embodiment is shown. At 401, the routine is started, and at 402, the outer join to be performed is identified. At 403, the full outer join is programmatically transformed into a union of a left outer join and a right outer join, minus the matched tuples from the right outer join. The rewritten SQL statement which implements the transformed full outer join is provided to the RDBMS' optimizer (not shown) at 404, and program execution continues at 405. Consider a full outer join with join condition P(T1, T2):

```
SELECT q1.c1, q1.c2, q2.c1, q2.c2
FROM T1 q1 FULL OUTER JOIN T2 q2
ON P(T1,T2)
```

> Having continued reference to FIG. 4, the transformation of this full outer join is programmatically implementable with the following set of SQL statements. These transform the full outer join into a union of a left outer join and a select query with a "not-exists" subquery, as follows:

```
SELECT q1.c1, q1.c2, q2.c1, q2.c2
FROM T1 q1 LEFT OUTER JOIN T2 q2
ON P(T1,T2)
UNION ALL
SELECT null, null, q2.c1, q2.c2
FROM T2 q2
WHERE NOT EXISTS (     SELECT 1
FROM T1 q1
WHERE P(T1,T2))
```

> The second operand of the union query returns T2 rows which do not match any rows in T1.

It is recognized that transforming a full outer join into a union query with a not-exists subquery may not provide optimally efficient computational performance. This is due to the fact that many subqueries are inherently slow to execute, especially in the MPP shared-nothing environment. Accordingly, the second preferred embodiment of the present invention presents a novel and computationally efficient full outer join with an arbitrary join condition capability.

Having reference to FIG. 5, this embodiment of the present invention also transforms the full join query into a union query of left outer join and right outer join, at 501-503. However, in order to ensure that the query returns the correct answer, for the right outer join all the matched rows are filtered out from 504-506. To indicate that the output from the right outer join is a matched row, a non-nullable column of the null-producing operand is added to the output of the right outer join at 504. The non-nullable column includes a primary key (key), and a row identifier (rid) or tuple identifier (tid). The value of these columns is set to null at 505 when there is not a match on the tuple-preserving operand by the definition of the outer join. By applying the "IS NULL" predicate after the right outer join at 506, all matched rows are removed from the answer set. The rewritten SQL statement which implements the transformed full outer join is provided to the RDBMS' optimizer (not shown) at 507, and program execution continues at 508.

Consider the following three queries:

```
I. SELECT q1.c1, q1.c2, q2.c1, q2.c2
FROM T1 q1 LEFT OUTER JOIN T2 q2
ON P(T1,T2)
UNION ALL
(SELECT q.c1, q.c2, q.c3, q.c4
FROM TABLE(SELECT q3.c1, q3.c2, q4.c1, q4.c2, q3.tid
FROM T1 q3 RIGHT OUTER JOIN T2 q4
ON P(T1,T2)) as q(c1, c2, c3, c4, tid)
WHERE q.tid IS NULL)
II. SELECT q1.c1, q1.c2, q2.c1, q2.c2
FROM T1 q1 LEFT OUTER JOIN T2 q2
ON P(T1,T2) UNION ALL
(SELECT q.c1, q.c2, q.c3, q.c4
FROM TABLE(SELECT q3.c1, q3.c2, q4.c1, q4.c2, q3.rid
FROM T1 q3 RIGHT OUTER JOIN T2 q4
ON P(T1,T2)) as q(c1, c2, c3, c4, rid)
WHERE q.rid IS NULL)
III. SELECT q1.c1, q1.c2, q2.c1, q2.c2
FROM T1 q1 LEFT OUTER JOIN T2 q2
ON P(T1,T2)
UNION ALL
(SELECT q.c1, q.c2, q.c3, q.c4
FROM TABLE(SELECT q3.c1, q3.c2, q4.c1, q4.c2, q3.key
FROM T1 q3 RIGHT OUTER JOIN T2 q4
ON P(T1,T2)) as q(c1, c2, c3, c4, key)
WHERE q.key IS NULL)
```

It will be appreciated that each of these three queries returns the same answer set.

The preceding approach is acceptable given a non-nullable column. FIG. 6 details another preferred embodiment useful where an outer join operand does not have any non-nulable column, for instance when the operand is a derived table, a column of constants could be added to achieve the same effect. For example, suppose both DT1 and DT2 are not base tables: i.e., their row identifiers were not readily avialable, and all columns were nullable. A full outer join for such a condition could then be rewritten as:

```
SELECT q1.c1, q1.c2, q2.c1, q2.c2
FROM DT1 q1 FULL OUTER JOIN DT2 q2
ON P(DT1,DT2)
```

which in turn can be implemented as:

```
SELECT q1.c1, q1.c2, q2.c1, q2.c2
FROM DT1 q1 LEFT OUTER JOIN DT2 q2
ON P(DT1,DT2)
UNION ALL
(SELECT q.c1, q.c2, q.c3, q.c4
FROM TABLE(SELECT q3.c1, q3.c2, q4.c1, q4.c2, q3.key
FROM TABLE(SELECT DT1.c1, DT1.c2, 1 FROM DT1)as q3(c1, c2, key)
RIGHT OUTER JOIN DT2 q4 ON P(DT1,DT2)) as q(c1,c2, c3, c4, key)
WHERE q.key IS NULL)
```

Cheng: Col. 5, lines 45-56 (actually, lines 43-56)

Given that the principles of the present invention implement a full outer join as a union of a left and a right outer join, a further optimization is accomplished by the final embodiment of the present invention by recognizing that all the matched rows in the right outer join are filtered out. This embodiment is implemented by using as a second operand of the union query a novel operator, the "ANTI-JOIN". The anti-join can be implemented as a right outer join coupled with an ISNULL predicate, or equivalently, it can be implemented directly using a new run time operator. Since computing all matched rows which will be eventually filtered out is computationally inefficient, the use of the anti-join presents significant savings in efficiency of implementation.

Cheng: Col. 11, lines 20-25

Given that the present invention implements a full outer join as a union of a left and a right outer join, the fourth preferred embodiment further optimizes the full outer join methodology taught herein by recognizing that all the matched rows in the right outer join are to be filtered out. In essence, the second operand of the union query is an anti-join, which is implemented as a right outer join coupled with an ISNULL predicate. Computing all matched rows which will be eventually filtered out can be very time consuming. Thus, one can implement the anti-join more

efficiently by an outer join algorithm with early out capability. This leads to the following query rewrite transformation.

Nothing in the description from Cheng set forth above teaches or suggests the independent claim limitations directed to "combining join predicates from a query with local predicates from each branch of one or more UNION ALL views referenced by the query," "analyzing the combined predicates," and "generating the join, except when the analyzing step indicates that the combined predicates are always FALSE and the join generates an empty result."

Instead, the description from Cheng set forth above merely describes a mechanism that enables a query implementing a full outer join with any join conditions, including an inequality predicate and disjunction, by equating a full outer join with a left outer join unioned with a right outer join less the matched tuples from the right outer join. In addition, the description from Cheng set forth above merely describes an anti-join that is implemented as a right outer join coupled with an ISNULL predicate.

However, these implementations by Cheng of a full outer join and anti-join do not perform the recited steps or functions of Applicants' independent claims. For example, nowhere does Cheng describe branches of UNION ALL views, combining join predicates from a query with local predicates from each branch of one or more UNION ALL views referenced by the query, for the purposes of analyzing the combined predicates, and then generating the join, except when the analysis of the combined predicates indicates that the combined predicates are always FALSE and the join generates an empty result.

Jou fails to overcome the deficiencies of Cheng. Recall that Jou was only cited against dependent claims 6-9, 19-22 and 32-35, and only on the basis that Jou teaches a select-list of the query containing a DISTINCT modifier, one or more aggregate functions and/or a GROUP BY clause, as well as a regrouping query block with a regrouping select-list, a GROUP BY clause, and another select-list used for distribution. Further, Applicants' attorney traverses these assertions and submits that the cited locations in Jou merely describe a select-list of a query containing a DISTINCT modifier, a subquery, aggregate functions, and a GROUP BY clause. However, at the cited location, Jou teaches nothing about creating a regrouping query block with a regrouping select-list, and a GROUP BY clause, and another select-list used for distribution.

Thus, even when combined, Cheng and Jou teach away from Applicants' invention. Moreover, the various elements of Applicants' claimed invention together provide operational

advantages over Cheng and Jou. In addition, Applicants' invention solves problems not recognized by Cheng and Jou.

Thus, Applicants submit that independent claims 1, 14, and 27 are allowable over Cheng and Jou. Further, dependent claims 2-13, 15-26, and 28-39 are submitted to be allowable over Cheng and Jou in the same manner, because they are dependent on independent claims 1, 14, and 27, respectively, and thus contain all the limitations of the independent claims. In addition, dependent claims 2-13, 15-26, and 28-39 recite additional novel elements not shown by Cheng and Jou.

V. Conclusion

In view of the above, it is submitted that this application is now in good order for allowance and such allowance is respectfully solicited. Should the Examiner believe minor matters still remain that can be resolved in a telephone interview, the Examiner is urged to call Applicants' undersigned attorney.

Respectfully submitted,

GATES & COOPER LLP
Attorneys for Applicants

Howard Hughes Center
6701 Center Drive West, Suite 1050
Los Angeles, California 90045
(310) 641-8797

Date: June 23, 2006

By: ______________________
Name: George H. Gates
Reg. No.: 33,500

GHG/

G&C 30571.293-US-01